UNIVERSITÀ DEGLI STUDI DI BERGAMO | Department of Management, Information and Production Engineering

INGEGNERIA DEL SOFTWARE: PROJECT PLAN

DEVELOPER:
SABRIN MAATOUG, MATRICOLA 1065576

1. INTRODUZIONE

Il software EasyDhack è finalizzato a migliorare la gestione di un bar-tabaccheria. Lo staff interno è costituito da un proprietario e dai dipendenti, mentre le figure esterne sono i fornitori. Attraverso il software, sarà garantita un'efficiente gestione del magazzino con riordino automatico/manuale dei prodotti e controllo scadenze, caricamento dei documenti di trasporto tramite semplice interfaccia e conseguente aggiornamento automatico delle giacenze.

2. PROCESS MODEL

Il bar non è un sistema che prevede continui e rapidi cambiamenti, tuttavia il proprietario deve essere in grado di interagire attivamente durante l'evoluzione del progetto, perciò la scelta più logica ricade sul modello RUP, intermedio tra agile e heavyweighted, che meglio si presta a questa situazione.
Ognuna delle quattro fasi del modello avrà una durata media di una settimana, al termine della quale verranno rilasciati i relativi documenti.

3. ORGANIZZAZIONE DEL PROGETTO

II progetto verrà svolto per la maggior parte individualmente ed avendo buona conoscenza sia del locale che del magazzino sarà possibile verificare sul luogo la funzionalità del software.

4. STANDARD E PROCEDURE

Come linguaggio di programmazione per la stesura del codice verrà utilizzato Java con Eclipse come IDE, mentre il database verrà simulato da un gruppo di file.
Per la modellazione verrà utilizzato starUML con tool REBEL per la generazione dello scheletro del codice.

5. MANAGEMENT ACTIVITIES

Il progetto verrà svolto seguendo una timeline con scadenze settimanali (vedi schedule), in cui si eseguirà un check degli obiettivi da raggiungere con eventuali modifiche. I documenti, i modelli e parti di codice terminati verranno caricati man mano nella cartella dedicata, mentre la consegna finale consisterà in un unico pdf contenente tutta la documentazione specificato nel readMe.

6. RISCHI

Il sistema verrà totalmente simulato, perciò non sono previsti rischi di compatibilità. Per quanto riguarda lo sviluppo, le tempistiche potrebbero risultare ristrette per una sola persona, perciò l'implementazione potrebbe non essere completata.

7. STAFFING

Il lavoro verrà svolto in maniera individuale per un periodo di circa un mese per un totale complessivo di almeno 40 ore.

8. METODI E TECNICHE

Requirements Engineering:

- Elicitation: metodo Tayloriano
- Requirements Specification: IEEE standard 830
- Verification&Validation: test plan

Requirement Analysys: Scala MoSCoW.

Design: IEEE standard 1471

9. QUALITY ASSURANCE

Il controllo della qualità del processo si baserà sullo standardISO9000 per la verifica del soddisfacimento dei requisiti.

10. WORK PACKAGES

Il lavoro sarà suddiviso nei nove workflows previsti dal modello RUP, che garantisce una divisione logica delle attività.

11. RISORSE

- Project plan: OpenOffice Writer
- Diagrammi dei modelli: starUML
- Generazione scheletro del codice: tool REBEL
- Implementazione e testing: JavaSE 16, EclipseIDE (2022-12 release)
- Documentazione: OpenOffice Writer
- Presentazione: OpenOffice Impress

12. BUDGET E SCHEDULE

Il lavoro verrà svolto su un Personal Computer, perciò non sono previsti costi economici. Lo schedule prevede cinque date di consegna:

- Project Plan: entro il 31/12/2022
- Specifica dei requisiti, specifica dell'architettura: entro il 10/01/2023
- Specifica del design, modelli: entro il 20/01/2023
- Beta release: entro il 31/01/2023
- Software release: entro il 05/02/2023

13. MODIFICHE

Gli aggiornamenti e le modifiche verranno gestite e monitorate tramite un repository github condiviso.

14. DISTRIBUZIONE

Tutta la documentazione, i modelli e il codice verranno distribuiti tramite repository github.